CLAIMS:

1. For a software application that uses a first interface related to a first piece of software code, a method of migrating the software application to allow the software application to use a second interface instead of the first interface, the method comprising:

creating a computer-readable mapping between the first interface and the second interface;

running the mapping through an auto-generator, wherein the auto-generator uses the mapping to automatically generate an interface wrapper;

replacing the first interface and the first piece of software code with the interface wrapper, thereby interposing the interface wrapper between the software application and the second interface;

wherein the interface wrapper allows the software application to communicate with the second interface instead of the first interface.

2. The method of claim 1, wherein the interface wrapper provides for at least one of (a) forward compatibility wherein the interface wrapper allows the software application to transparently communicate with the second interface and (b) backward compatibility wherein the interface wrapper allows the second interface to transparently communicate with the software application.

3. The method of claim 2, wherein for forward compatibility, creating the computer-readable mapping comprises creating a mapping from the first interface to the second interface.

4. The method of claim 2, wherein for backward compatibility, creating the computer-readable mapping comprises creating a mapping from the second interface to the first interface.

5. The method of claim 1, wherein the auto-generator, in an object-oriented environment, is software comprising the following instructions:

a) for each class in the first interface

a.1) from all classes to be included in the interface wrapper mapped from the class in the first interface, select a master class to hold handles to all other classes in the interface wrapper mapped from the class in the first interface;

b) for each class to be included in the interface wrapper as set out in the computer-readable mapping

b.1) if the class to be included in the interface wrapper is the master class, initialize all the handles in the master class;

b.2) for each attribute in the class to be included in the interface wrapper

b.2.1) add code from the computer-readable mapping related to the attribute to the class to be included in the interface wrapper;

b.3) for each method in the class to be included in the interface wrapper

b.3.1) add code from the computer-readable mapping related to the method to the class to be included in the interface wrapper.

6. The method of claim 1, wherein the auto-generator, in an object-oriented environment, is software comprising the following instructions

for each class to be included in the interface wrapper as set out in the computer-readable mapping:

a) define the class to be included in the interface wrapper

b) for each class from the first interface mapped to the class to be included in the interface wrapper as set out in the computer readable mapping:

b.1) if the class from the first interface is mapped only to the class to be included in the interface wrapper

b.1.1) add a member for the class from the first interface to the class to be included in the interface wrapper;

b.1.2) add construction of the member to all constructors in the class to be included in the interface wrapper;

b.2) else if the class to be included in the interface wrapper is the first of all classes in the interface wrapper mapped from the class from the first interface as set out in the computer-readable mapping:

b.2.1) designate the class to be included in the interface wrapper as a master class to hold handles to all other classes in the interface wrapper mapped from the class in the first interface;

b.2.2) add a member for the class from the first interface to the class to be included in the interface wrapper;

b.2.3) add construction of the member to all constructors in the class to be included in the interface wrapper;

b.2.4) for each other class in the interface wrapper mapped from the class from the first interface

b.2.4.1) add a member for the other class in the interface wrapper to the master class;

b.2.4.2) add construction of the other class in the interface wrapper to all constructors in the master class;

b.2.4.3) add a call to initialize the member wherein the member will know that the class to be included in the interface wrapper is the master class in all constructors of the other class in the interface wrapper;

b.2.4.4) add a method to the other class in the interface wrapper to retrieve

the member by a type name of the other class;

b.3) else

b.3.1) add a member for the master class to the class to be included in the interface wrapper;

b.3.2) add a method to the class to be included in the interface wrapper to allow the member to be initialized to point to the master class;

b.3.3) add a method to the class to be included in the interface wrapper to retrieve the member;

c) for each attribute in the class to be included in the interface wrapper

c.1) add code from the computer-readable mapping related to the attribute to the class to be included in the interface wrapper;

d) for each method in the class to be included in the interface wrapper

d.1) add code from the computer-readable mapping related to the method to the class to be included in the interface wrapper.

7. The method of claim 1, wherein the software application, the first interface and the second interface are written in the Java language.

8. The method of claim 1, wherein the computer-readable mapping is written in a language selected from one of XML and UML.

9. The method of claim 1, wherein the auto-generator is software written in a form selected from one of XSL templates, bean script and XDoclet.

10. The method of claim 1, wherein the software application is migrated from the first interface to the second interface without modifying the software application.

11. A system adapted to assist with the migration of a software application from a first interface to a second interface, the system comprising a processing platform and a computer-readable medium comprising auto-generation software, the system being adapted to receive a computer-readable mapping from the first interface to the second interface, the processing platform being adapted to execute instructions of the auto-generation software to process the mapping to produce an interface wrapper wherein the interface wrapper allows the software application to transparently communicate with the second interface.

12. The system of claim 11, wherein the instructions of the auto-generation software comprises the following instructions:

a) for each class in the first interface

a.1) from all classes to be included in the interface wrapper mapped from the class in the first interface, select a master class to hold handles to all other classes in the interface wrapper mapped from the class in the first interface;

b) for each class to be included in the interface wrapper as set out in the computer-readable mapping

b.1) if the class to be included in the interface wrapper is the master class, initialize all the handles in the master class;

b.2) for each attribute in the class to be included in the interface wrapper

b.2.1) add code from the computer-readable mapping related to the attribute to the class to be included in the interface wrapper;

b.3) for each method in the class to be included in the interface wrapper

b.3.1) add code from the computer-readable mapping related to the method to the class to be included in the interface wrapper.

13. The system of claim 11 wherein the instructions from the auto-generation software comprises the following instructions:

-for each class to be included in the interface wrapper as set out in the computer-readable mapping:

a) define the class to be included in the interface wrapper

b) for each class from the first interface mapped to the class to be included in the interface wrapper as set out in the computer readable mapping:

b.1) if the class from the first interface is mapped only to the class to be included in the interface wrapper:

b.1.1) add a member for the class from the first interface to the class to be included in the interface wrapper;

b.1.2) add construction of the member to all constructors in the class to be included in the interface wrapper;

b.2) else if the class to be included in the interface wrapper is the first of all classes in the interface wrapper mapped from the class from the first interface as set out in the computer-readable mapping:

b.2.1) designate the class to be included in the interface wrapper as a master class to hold handles to all other classes in the interface wrapper mapped from the class in the first interface;

b.2.2) add a member for the class from the first interface to the class to be included in the interface wrapper;

b.2.3) add construction of the member to all constructors in the class to be included in the interface wrapper;

b.2.4) for each other class in the interface wrapper mapped from the class from the first interface

b.2.4.1) add a member for the other class in the interface wrapper to the master class;

b.2.4.2) add construction of the other class in the interface wrapper to all constructors in the master class;

b.2.4.3) add a call to initialize the member wherein the member will know that the class to be included in the interface wrapper is the master class in all constructors of the other class in the interface wrapper;

b.2.4.4) add a method to the other class in the interface wrapper to retrieve the member by a type name of the other class;

b.3) else

b.3.1) add a member for the master class to the class to be included in the interface wrapper;

b.3.2) add a method to the class to be included in the interface wrapper to allow the member to be initialized to point to the master class;

b.3.3) add a method to the class to be included in the interface wrapper to retrieve the member;

c) for each attribute in the class to be included in the interface wrapper

c.1) add code from the computer-readable mapping related to the attribute to the class to be included in the interface wrapper;

d) for each method in the class to be included in the interface wrapper

d.1) add code from the computer-readable mapping related to the method to the class to be included in the interface wrapper.

14. A computer-readable medium containing instructions for automatically generating an interface wrapper to facilitate migration of a software application from a first interface to a second interface, wherein, given a computer-readable mapping from the first interface to the second interface, the instructions comprise:

a) for each class in the first interface

a.1) from all classes to be included in the interface wrapper mapped from the class in the first interface, select a master class to hold handles to all other classes in the interface wrapper mapped from the class in the first interface;

b) for each class to be included in the interface wrapper as set out in the computer-readable mapping

b.1) if the class to be included in the interface wrapper is the master class, initialize all the handles in the master class;

b.2) for each attribute in the class to be included in the interface wrapper

b.2.1) add code from the computer-readable mapping related to the attribute to the class to be included in the interface wrapper;

b.3) for each method in the class to be included in the interface wrapper

b.3.1) add code from the computer-readable mapping related to the method to the class to be included in the interface wrapper.

15. A computer-readable medium containing instructions for automatically generating an interface wrapper to facilitate migration of a software application from a first interface to a second interface, wherein, given a computer-readable mapping from the first interface to the second interface, the instructions comprise:

-for each class to be included in the interface wrapper as set out in the computer-readable mapping,

a) define the class to be included in the interface wrapper

b) for each class from the first interface mapped to the class to be included in the interface wrapper as set out in the computer readable mapping

b.1) if the class from the first interface is mapped only to the class to be included in the interface wrapper

b.1.1) add a member for the class from the first interface to the class to be included in the interface wrapper;

b.1.2) add construction of the member to all constructors in the class to be included in the interface wrapper;

b.2) else if the class to be included in the interface wrapper is the first of all classes in the interface wrapper mapped from the class from the first interface as set out in the computer-readable mapping

b.2.1) designate the class to be included in the interface wrapper as a master class to hold handles to all other classes in the interface wrapper mapped from the class in the first interface;

b.2.2) add a member for the class from the first interface to the class to be included in the interface wrapper;

b.2.3) add construction of the member to all constructors in the class to be included in the interface wrapper;

b.2.4) for each other class in the interface wrapper mapped from the class from the first interface

b.2.4.1) add a member for the other class in the interface wrapper to the master class;

b.2.4.2) add construction of the other class in the interface wrapper to all constructors in the master class;

b.2.4.3) add a call to initialize the member wherein the member will know that the class to be included in the interface wrapper is the master

class in all constructors of the other class in the interface wrapper;

b.2.4.4) add a method to the other class in the interface wrapper to retrieve the member by a type name of the other class;

b.3)else

b.3.1) add a member for the master class to the class to be included in the interface wrapper;

b.3.2) add a method to the class to be included in the interface wrapper to allow the member to be initialized to point to the master class;

b.3.3) add a method to the class to be included in the interface wrapper to retrieve the member;

c) for each attribute in the class to be included in the interface wrapper

c.1) add code from the computer-readable mapping related to the attribute to the class to be included in the interface wrapper;

d) for each method in the class to be included in the interface wrapper

d.1) add code from the computer-readable mapping related to the method to the class to be included in the interface wrapper.